A highly parallel code for strongly coupled fluid-transport equations

Song, Weiyan; Wubs, Fred W.; Thies, Jonas

Published in:
Proceedings of the World Congress on Computational Mechanics

Publication date:
2014

11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
E. Oñate, J. Oliver and A. Huerta (Eds)

A HIGHLY PARALLEL CODE FOR STRONGLY COUPLED FLUID-TRANSPORT EQUATIONS

Weiyan Song*, Fred W. Wubs* and Jonas Thies †

* Johann Bernoulli Institute for Mathematics and Computing Science, University of Groningen
P.O.Box 407, 9700 AK Groningen, The Netherlands
E-mail: W.Song@rug.nl, web page: http://www.math.rug.nl/cmnm

† Department for Simulation and Software Technology, German Aerospace Center
Linder Höhe, 51147 Cologne, Germany
E-mail: jonas.thies@dlr.de, URL: http://www.dlr.de/sc

Key words: Multiphysics Problems, High Performance Computing, Numerical Methods, Multilevel ILU

Abstract. We developed a finite volume package FVM and a solver HYMLS, both based on elements of the Trilinos EPETRA-package (see http://trilinos.sandia.gov/). HYMLS is a linear system solver for steady state incompressible Navier-Stokes equations coupled to transport equations in 2 and 3D [1, 2, 3]. We constructed recently a multilevel variant of it, which makes it possible to solve 3D problems of over 10 million unknowns quickly on a parallel computer. The behavior of the method is very much like that of multigrid methods. In fact one could see it as the father of the multigrid method. The solver is very robust. For the problem described in [4], it allowed a quick increase in the Reynolds number to get into the interesting region around Re=2000. Here we will show the performance of the method on the Rayleigh-Bénard convection in a cube, with six no-slip walls [5]. Also here we employ HYMLS to solve the linear systems resulting from a Cayley transform of the generalized eigenvalue problem.

1 INTRODUCTION

Many flow problems deal with transport of matter and/or heat. This constitutes a challenging multiphysics problem if the transported entity also influences the flow, leading to a two-way coupling. From a computing efficiency view point, it is best to treat the associated equations in a coupled manner [6]. Employing a domain decomposition approach, all the unknowns related to one domain should be in the memory of the node which treats that part and communication should be avoided as much as possible during the construction of the right-hand side, the construction of the Jacobian matrix and the solution process. For this first version, we have chosen to use structured grids and finite

volume discretizations. For the incompressible Navier-Stokes equation, until now, we use the C-grid staggering. Our implementation is matrix oriented instead of function oriented. Before computation we compute and store all stencils needed in the code. From the nonlinear terms, which are here bilinear forms, we store the constituting operators also in the form of stencils. The Jacobian matrix and the right-hand side are now constructed from products of stencils and vectors. To solve the nonlinear equations we use the Trilinos NOX package with our in house developed package HYMLS to solve the linear systems.

In this paper, we apply our method to two well known CFD benchmark problems: flow in a 3D lid-driven cavity and Rayleigh-Bénard convection in a cube. For both cases, we investigate the the first critical point. To study the stability of the solutions we determine the eigenvalues using the ANASAZI-package, which contains a generalized version of the Arnoldi method. Also here, we employ HYMLS to solve the linear systems resulting from a Cayley transform of the generalized eigenvalue problem. In Section 2, we present the problem formulation. Section 3 gives a full description about the method employed. In Section 4, we presents the numerical experiments for the above two problems. Conclusions are given in Section 5.

2 PROBLEM FORMULATION

Our aim is to study the dynamics of fluid flow problems. In first instance we want to consider the ones related to incompressible flow. So in general we like to solve the Navier-Stokes equations combined with a number of transport eqautions:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} &= -\nabla p + \mu\Delta\mathbf{u} + \mathbf{f}(\phi) \\ \nabla\cdot\mathbf{u} &= 0 \\ \frac{\partial \phi}{\partial t} + \mathbf{u}\cdot\nabla\phi &= \kappa\Delta\phi + \mathbf{g}(\mathbf{u}) \end{aligned} \tag{1}$$

where ϕ is a vector of components that are transported with the flow. For instance, one could think of transport of energy, material, etc. In many cases there is interaction between the flow and the components that are transported. This is represented by the functions $\mathbf{f}$ and $\mathbf{g}$. As an example we treat in this paper the 3D lid-driven cavity and the Rayleigh-Bénard problem.

2.1 3D lid-driven cavity

We consider the flow of an incompressible Newtonian fluid in a 3D cavity with edges of length L (see Fig. 1), the fluid motion is driven by a moving lid on the top with constant velocity U. The boundary conditions are no-slip ($\mathbf{u} = 0$) on all the walls except for the top moving lid. The flow region is defined in the dimensionless Cartesian coordinates x, y, and z each of which varies from -0.5 to 0.5. The governing equations for this problem are the incompressible Navier-Stokes equations, which can be put into dimensionless form in

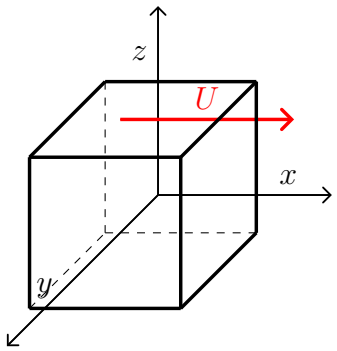

Figure 1: Geometry of cavity with moving lid on the top

Cartesian coordinates with the components of the velocity vector given by $\mathbf{u} = (u, v, w)$:

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \frac{1}{Re} \Delta \mathbf{u} \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned} \tag{2}$$

where Re is the Reynolds number.

2.2 Rayleigh-Bénard convection

Here, convection of a Boussinesq fluid in a cubic cavity with six no-slip walls is considered, where the bottom is kept at a higher temperature than the top of the cavity. The field equations are given by

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \Delta \mathbf{u} + Ra\, T \mathbf{e}_z \\ \nabla \cdot \mathbf{u} &= 0 \\ \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T &= \frac{1}{Pr} \Delta T + \frac{1}{Pr} \mathbf{u}_z \end{aligned} \tag{3}$$

where Ra is the Rayleigh number and Pr is the Prandtl number. The temperature T in these equations is the perturbation from a steady pure conduction temperature profile. So the temparature in these equations is prescribed to be zero at bottom and top. The lateral walls of the cavity are insulated walls so $\partial T/\partial n = 0$. Furthermore, at all walls the velocity is set to zero.

3 DESCRIPTION OF METHODOLODY

In this section we explain the discretization and the algorithms we employed. In general we will take the incompressible Navier Stokes equation as an example, the inclusion of transport equations is straightforward.

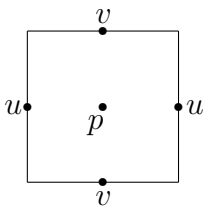

Figure 2: Positioning of velocity (u, v) and pressure (p) in the C-grid.

3.1 Symmetry preserving Finite Volume Discretization

For the discretization we write the continuous equations in the following form

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} &= -\mathcal{N}(\mathbf{u},\mathbf{u}) + \frac{1}{Re}\mathcal{L}\mathbf{u} - \nabla p \\ 0 &= \nabla \cdot \mathbf{u} \end{aligned}$$

Here the equations are in conservative form which is the usual starting point for the finite volume discretization. Now it can be shown [7] that on closed domains it holds that $\int_\Omega \mathbf{u}\mathcal{N}(\mathbf{u},\hat{\mathbf{u}})d\Omega = 0$ for any divergence free $\hat{\mathbf{u}}$. Hence dissipation of kinetic energy in a domain enclosed by walls can only occur by diffusion. We like to preserve this property in the discretization in order to preclude artificial diffusion.

Note also that $\mathcal{N}(\mathbf{u},\hat{\mathbf{u}})$ is a bilinear form which in terms of linear operators $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ can be written as $\mathcal{C}((\mathcal{A}\mathbf{u})(\mathcal{B}\hat{\mathbf{u}}))$. We can use this structure in the discretization and in the implementation.

We discretize the equations here with a second-order symmetry-preserving finite volumes on C-grid (see Fig. 2) with a parameter to enable stretching towards boundaries. The space discretized equations assume the form

$$M\frac{d\mathbf{u}}{dt} = -N(\mathbf{u},\mathbf{u}) + \frac{1}{Re}L\mathbf{u} - Gp \tag{4}$$

$$0 = D\mathbf{u} \tag{5}$$

where here $\mathbf{u}$ and p have turned into vectors. Now we can write

$$N(\mathbf{u},\hat{\mathbf{u}}) = N_1(\mathbf{u})\hat{\mathbf{u}} = N_2(\hat{\mathbf{u}})\mathbf{u} \tag{6}$$

where in MATLAB notation $N_1(\mathbf{u}) = C{*}diag(A\mathbf{u}){*}B$ and $N_2(\hat{\mathbf{u}}) = C{*}diag(B\hat{\mathbf{u}}){*}A$ where A, B and C are discretizations of the linear operators with the same names mentioned above. The system with the Jacobian, which typically has to be solved in a Newton step is of the form

$$\begin{pmatrix} -N_1(\mathbf{u}) - N_2(\mathbf{u}) + \frac{1}{Re}L & -G \\ D & O \end{pmatrix} \begin{pmatrix} \Delta\mathbf{u} \\ \Delta p \end{pmatrix} = - \begin{pmatrix} f_\mathbf{u} \\ f_p \end{pmatrix}$$

Observe that the matrix has a linear part consisting of diffusion, gradient and divergence operators and 2 nonlinear parts. Note that if we skip one of these parts from the Jacobian, then ,due to (6), when multiplying by $[\mathbf{u}, p]^T$ we get the right-hand side of (4).

3.2 Program structure

Our FVM/HYMLS package is based on Trilinos. It consists of two parts (i) FVM in which the user can define his problem and (ii) a continuation program. FVM is written in FORTRAN90 and does not contain any MPI. The continuation program is written in C++ and is heavily using data structures given in the Trilinos Epetra package.

FVM To facilitate the use of the program the application scientist has to create a number of basic routines, e.g. for the computation of the right-hand side and the Jacobian matrix. These routines can be written in FORTRAN90 and the interface is prescribed. The essential routines are the following.

Initialization Based on (i) x, y and z coordinates in each direction, (ii) the initial solution corresponding to the domain and (iii) a mask array determining the geometry and boundary conditions on the domain, we build the stencil arrays corresponding to second-order accurate Finite Volume Discretizations and the bilinear form.

Compute Jacobian matrix Using the stencils for the bilinear form, create a stencil for $N_1(\mathbf{u})$ and $N_2(\mathbf{u})$ (see 6). Construct the Jacobian matrix in CSR format from this.

Compute right-hand side Using the stencils for the bilinear form, create a stencil for $N_1(\mathbf{u})$. Together with stencils for linear part, current solution and forcing compute the right-hand side.

Compute mass matrix Since in our case the mass matrix is diagonal, we just compute the diagonal entries.

Continuation program The continuation program is written in C++ and makes use of the Epetra data structures. A short overview of the parallel continuation algorithm is given below.

- Initialization
  - Partitioning of the domain. Based on this, two maps are generated: one for overlapping domains and one for non-overlapping domains.
  - Initialize solution on the overlapping domains
  - Compute stencils for linear and bilinear form for each overlapping domain (call FVM:Initialization)

- Continuation using LOCA
  - Compute solution on nonoverlapping domains using the Newton method (NOX).
    - Compute right-hand side and Jacobian matrix on overlapping domains (FVM:Compute right-hand side, Compute Jacobian matrix). Next the computation is going on on the non-overlapping parts of these domains.
    - Solve linear system using HYMLS. Solution is found on the non-overlapping domains
    - Solutions are copied to overlapping domains.
  - Eigenvalue computation using ANASAZI
    - Compute Jacobian and Mass matrix (FVM:Compute Jacobian matrix, Compute mass matrix)
    - Transformed Matrix is computed and system is solved by HYMLS.

3.3 HYMLS

We consider the problem of solving the equations

$$Kx = b, \tag{7}$$

where $K \in R^{(n+m)\times(n+m)}$ $(n \geq m)$ is a saddle point matrix that has the form

$$K = \begin{pmatrix} A & G \\ G^T & 0 \end{pmatrix}, \tag{8}$$

with $A \in R^{n\times n}$, $G \in R^{n\times m}$. For the Stokes problem discretized on a C-grid (Fig. 2), K is a so-called $\mathcal{F}$-matrix (A is symmetric positive definite and G has row sum 0 and at most two entries per row [8]).

In [1] a direct method for the solution of $\mathcal{F}$-matrices was proposed. It reduces fill and computation time while preserving the structure of the equations during the elimination. A hybrid direct/iterative method based on this approach was presented in [2, 9]. It has the advantage that the ordering it defines for the matrix exposes parallelism on each level: all the subdomain matrices can be factored independently using sequential sparse direct solvers, and the Schur complement can be constructed with a minimal amount of communication in an assembly process. The difficult task of parallel preconditioning is aided by the algorithm, which yields a block-diagonal preconditioner with dense blocks and a significantly reduced sparse linear system. The ingredients of the associated incomplete LU factorization are the following.

1. Perform a non-overlapping domain decomposition. These domains are typically small, e.g. edge length 4, and are chosen independent of the partioning of the computational domain for parallelization.

2. Detect the velocity separators associated to this domain decomposition.

3. Pick for every subdomain one pressure unknown to be kept in the Schur complement.

4. Eliminate all interior variables of the subdomains and construct the Schur complement for the velocity separators and the selected pressure unknowns in 3.

5. Perform a Householder transformation on each separator. This decouples most of the velocity unknowns from the remaining pressure unknowns.

6. Identify V_Σ unknowns (separator velocities that still connect to two pressures).

7. Drop all connections between non-V_Σ unknowns and V_Σ unknowns, and between non-V_Σ unknowns in different separator groups. The resulting matrix is block-diagonal with the 'reduced Schur complement' defined by the V_Σ and pressure unknowns.

8. Repeat the process on the 'reduced Schur complement' till the number of levels specified.

9. Make a sparse direct factorization on the last Schur-complement

In [2] it is shown that for two levels the amount of iterations is independent of the mesh size. Since we are just repeating the process on the Schur complement it will be straightforward to show that also for a fixed number of levels the amount of iterations is independent of the mesh size. However with increasing number of levels the number of iterations increases; it does however only very mildly and in a monotonous way. The latter is due to the robustness of the method. The computational complexity depends on the number of iterations and the size of the last Schur-complement. This size increases with the problem size if the number of levels is fixed. By increasing the number of levels it drops significantly.

4 NUMERICAL EXPERIMENTS

4.1 3D lid-driven cavity

In our continuation program we set the Reynolds number as our continuation parameter. Starting from a small number, e.g., $Re = 1$, we let LOCA increase the Reynolds number withs step sizes 500 to $Re = 1900$ meanwhile computing the eigenvalues, in order to check the stability of the solution. Velocities are given in Fig. 3 and the results have second-order accuracy. For Reynolds numbers upto 1900, the real parts of the eigenvalues are all negative, that means that the steady flow is stable. When we increase the Reynolds number from 1900 to 2000, we observe that a conjugate pair of eigenvalues is crossing the imaginary axis, hence the flow loses its stability. Feldman and Gelfgat[4] were the first to show these results and found the critical Reynolds number to be 1927. They used a time

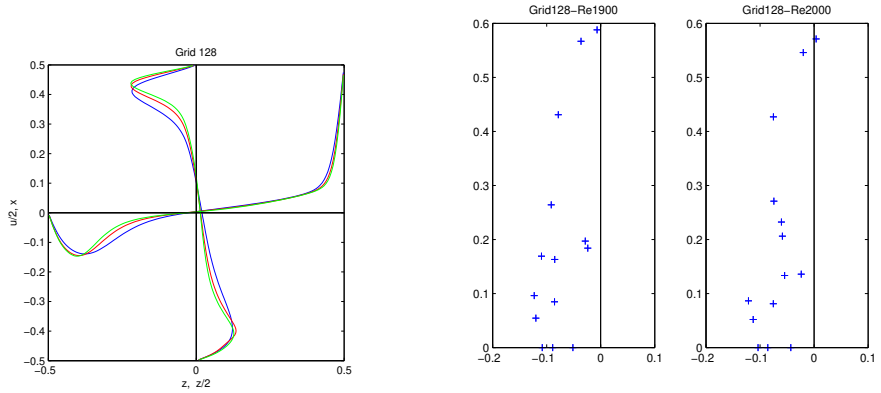

Figure 3: Steady state flow profiles obtained for grid 128^3 for $Re = 1000(b)$, $Re = 1500(r)$ and $Re = 1900(g)$. $u/2$ and $w/2$ velocity components along centerlines (0,0,z) and (x,0,0), respectively (left); Eigenvalues closest to zero at Re=1900 and Re=2000 (right).

Table 1: Eigenvalues obtained on different grids at $Re = 1900$ and $Re = 2000$

Re	λ_{32}	λ_{64}	λ_{128}
1900	$-53.05 + 0.468i$	$-22.28 + 0.550i$	$-6.762 + 0.588i$
2000	$-42.35 + 0.463i$	$-10.73 + 0.545i$	$3.901 + 0.571i$
Re_c	2395	2093	1963

dependent code to do so. In Table 1, we show our results, and add that from an extrapolation based on the second-order behavior of the error we expect the critical Reynolds number to be about 1920. Kuhlmann and Albensoeder [10] however did very accurate computations using a spectral method and found 1919.5. They computed this also by time integration of the equations for various Reynolds numbers. They also studied the subcritical behavior of this bifurcation in more detail and found that already at Re=1921 complicated dynamics occurs by interfering non-symmetric modes, i.e. modes that do not adopt the mirror symmetry around the plane $y = 0$.

4.2 Rayleigh-Bénard convection

Note that the no-flow solution is a solution of the equations for all Rayleigh and Prandtl numbers. The transition from conduction, i.e. the no-flow case, to convection starts when the Rayleigh number is increased beyond a critical value Ra_c, i.e. where the no-flow solution becomes unstable. It is in fact the Rayleigh number for which the Jacobian associated to the linear part of (3) becomes singular. This critical Rayleigh number is independent of the Prandtl number, but depends on the boundary conditions. For the box

Table 2: Eigenvalues, the last iteration number of Arnoldi process and Ra_c at different grids in 2D and 3D cases

Grid		iterations	Ra_c
16	2D	27	2517
	3D	54	3283
32	2D	39	2567
	3D	60	3360
64	2D	55	2581
	3D	89	3381
128	2D	59	2584
	3D	129	3387

geometry in 2D, the numerical value of Ra_c is 657.5 for the free-slip boundary condition and 1707 for the no-slip boundary condition [11].

Since the Rayleigh number appears as a coefficient in the equations, we can transform the determination of the critical Rayleigh number to an eigenvalue problem. So in stead of $det(J(Ra)) = 0$, we can write it as $det(Ra\ J_1 - J_2) = 0$. By a similarity transformation, in fact replacing T by $\sqrt{Ra}T$ we can transform this into a generalized eigenvalue problem for $\sqrt{Ra}$ in which both matrices are symmetric $\sqrt{Ra}\hat{J}_1 - \hat{J}_2$. Moreover, this is an eigenvalue problem on the space of divergence free velocities and on this space $\hat{J}_2$ is definite. Hence, all the eigenvalues $\sqrt{Ra}$ will be real. For the implementation we just adapted the computation of the Jacobian to give us two matrices instead of one. The eigenvalue problem in itself is similar to what we already had for computations of the stability of solutions where the Jacobian and the mass matrix are needed.

For the Anazasi eigen solver we choose "Shift and invert", with shift zero to find the eigenvalues. HYMLS is used to solve the according linear system. Table 3 presents the results. By this approach, we can directly locate $Ra_c = 3387$ for the 3D case and $Ra_c = 2584$ for the 2D case, which is the same with that found by Gelfgat [5].

One observes that the Rayleigh numbers are converging nicely with the grid refinement. Moreover the number of iterations increases monotonously and only very mildly with grid refinement for both the 2 and 3D case. This shows the robustness of the preconditioner.

To study the *weak scalability* of the method on a parallel computations, we perform computations on a 2D case. The computer used is an opteron cluster with inifiniband connection between the nodes; every node contains 12 cores. The results are shown in Table 3. Again we have a square cavity, with equal number of grid points in each direction. NL and NP denote the number of levels in the preconditioner and the number of cores used, respectively. Iters represents the number of GMRES iterations performed by HYMLS in the last iteration of the Arnoldi process. LU means the time the HYMLS

Table 3: Scalability test on 2D case

Grid	NL	NP	Iters	LU fact.	Solve
64	3	16	65	0.32	0.20
128	3	32	72	0.72	0.48
256	3	32	76	2.85	2.58
512	3	32	77	38.40	12.30
1024	3	128	80	728.00	54.40
1024	4	128	122	40.00	33.00
2048	4	128	124	56.00	104.00

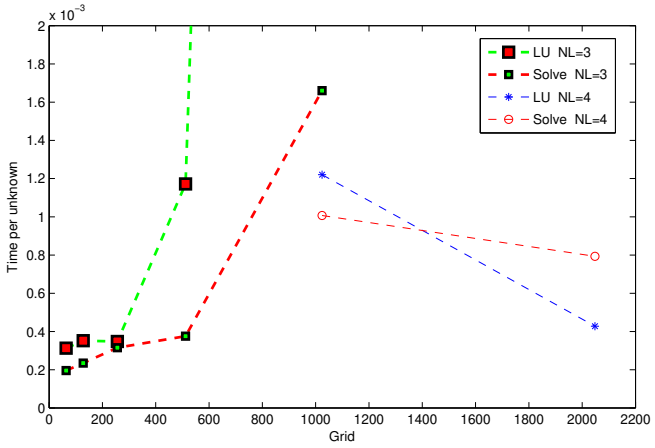

Figure 4: Time per unknown for computation of LU factorization and Solve

LU factorization takes and Solve means the time the linear solver takes, stopping criterion is $1e-8$.

We observe that the number of iterations is reaching a limit if the number of levels (NL) are kept constant. We have proven this behavior in [2]. However, if we keep NL fixed then the size of the last Schur-complement increases on refinement by a factor 4, and the time of its factorization will dominate the computations. Therefore, we use an additional level and since the separator size in this case is 4, the size of the last Schur-complement will be 16 times smaller. This explains the large drop in computation time wen we go from 3 to 4 levels keeping the same grid (1024).

For the solve time, we see a similar but less pronounced behavior, since solving with a rather full L and U-factor is much cheaper than creating them. In Fig. 4, we depicted the time which is needed per unknown to make an LU factorization and the time to solve the equations. These are obtained from the table by computing (NP*LUfact)/(4*Nxˆ2), so

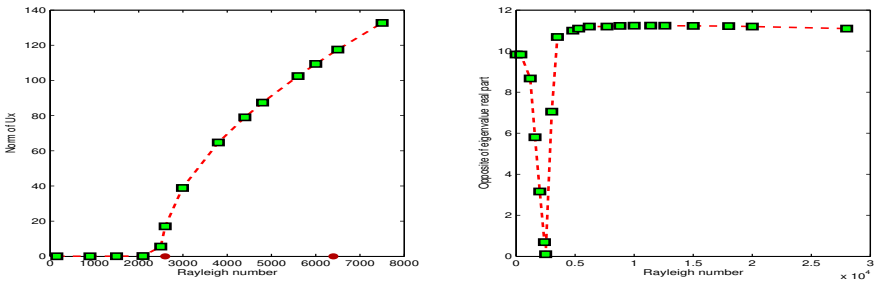

Figure 5: Norm of velocity in x direction U_x (left, two points on x-axis indicate the first two critical Rayleigh numbers) and $-\lambda_{real}$ (right) w.r.t Rayleigh number for 2D case

the total CPU time consumed divided by the total number of unknowns. So in case of an optimal speedup and gridindependent convergence, one would see an horizontal line. In this case, we see clearly the effect of keeping the number of levels constant. The interesting point is that apart from the mentioned effect the time per unknown for the factorization does not increase strongly, a factor 2 for the last problem which is about 1000 times bigger than the first and a factor 4 for the solve. These computations were carried out during normal operation of the cluster, so there may be effects due to distribution of the program over the various nodes, time sharing, cash effectcs, etc. In view of this we consider the results obtained as very acceptable.

Until now, computations on the trivial solution branch were presented. One way to get away from the trivial branch to the *non-trivial stable branch*, is to add a perturbation to the system, just before we are meeting a bifurcation point. After switching to the non-trivial solution, the perturbation can be turned off again. As perturbation we add the the right-hand side of the equations $\varepsilon M v$, where $\varepsilon = 1.0$, M is the mass matrix and v is the eigenvector obtained at $Ra = Ra_c$. In Fig. 5, the norm of the velocity and the opposite of the real part of the largest real eigenvalue $-\lambda_{real}$ are depicted with respect to the Rayleigh number. From the real part of eigenvalues, we can see that the non-trivial branch is stable for all shown Rayleigh numbers. Similar behavior is found in 3D case.

5 CONCLUSIONS

We have discussed an implementation of a package for analysing the dynamics of fluid flows coupled to transport equations. In this implementation we solve the coupled equations at once. This is quite natural when considering nonoverlapping domain decomposition using separators (in fact leading to a kind of Nested Dissection approach). The problem that leads to fuller and fuller Schur-complement matrices is solved by the transform-and-drop approach used in HYMLS. This allows for a significant reduction of the number of unknowns on the separators, which is similar to an aggressive coarsening in multigrid context. At the same time we can keep the nice properties like symmetry and positiviness of matrices.

This robust approach also admits a parallel implementation, which we performed using Trilinos packages. Experiments in this paper show an acceptable weak scaling. These results were obtained on a time sharing computer. Improvements of the implementation are underway.

We have also shown that the results for the two benchmark problems obtained by the hybrid solver FVM/HYMLS are consistent with those in related literature. From this we conclude that the FVM/HYMLS solver makes it possible to perform bifurcation analysis and steady state computations on large CFD problems. It is easy to extend it with more physics, temperature, salt etc..

REFERENCES

[1] A.C. de Niet and F.W. Wubs. Numerically stable LDL^T factorization of F-type saddle point matrices. *IMA Journal of Numerical Analysis*, **29**(1), 208-234, 2009.

[2] F.W. Wubs and J. Thies, A robust two-level incomplete factorization for (Navier-)Stokes saddle point matrices, *SIAM J. Matrix Anal. Appl.*, **32**, 1475 - 1499, 2011.

[3] Jonas Thies and Fred Wubs, Design of a Parallel Hybrid Direct/Iterative Solver for CFD Problems. *Proceedings 2011 Seventh IEEE International Conference on eScience*, 5-8 December 2011, Stockholm, Sweden, pp 387 -394, 2011.

[4] Yuri Feldman and Alexander Yu. Gelfgat, Oscillatory instability of a three-dimensional lid-driven flow in a cube, *Phys. Fluids*, **22**, 2010.

[5] Alexander Yu. Gelfgat, Different Modes of RayleighBenard Instability in Two- and Three-Dimensional Rectangular Enclosures, *J. Comput. Phys.*, **156**, 300324, 1999.

[6] David E. Keyes et al., Multiphysics simulations: Challenges and opportunities, *International Journal of High Performance Computing Applications*, **27**(1), 4-83, 2013.

[7] R.W.C.P. Verstappen and A.E.P. Veldman, Symmetry-preserving discretization of turbulent flow, *J. Comput. Phys.*, **187**, 343-368, 2003

[8] M.Tůma, A note on the LDL decomposition of matrices from saddle-point problems, *SIAM J. Matrix Anal. Appl.*, **23**, 2002.

[9] J. Thies and F.W. Wubs, A robust parallel ILU solver with grid-independent convergence for the coupled steady incompressible Navier-Stokes equations, *Proc. ECCOMAS CFD 2010*, J. C. F. Peireira and A.Sequeira, Eds., 2010,

[10] Hendrik C. Kuhlmann and Stefan Albensoeder, Stability of the steady three-dimensional lid-driven flow in a cube and the supercritical flow dynamics, *Phys. Fluids*, **26**, 2014.

[11] P.Manneville, *Instabilities, Chaos, and Turbulence*. Imperial College Press, London, 2004.